```
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPPP
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEE                    TTT        PPP         PPP
UUU         UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP
UUU         UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP
UUU         UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP
UUU         UUU  EEE                    TTT        PPP
UUU         UUU  EEE                    TTT        PPP
UUU         UUU  EEE                    TTT        PPP
UUU         UUU  EEE                    TTT        PPP
UUU         UUU  EEE                    TTT        PPP
UUU         UUU  EEE                    TTT        PPP
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP
```

FILEID**UETIDBDEF

```
UU      UU  EEEEEEEEEE  TTTTTTTTTT    IIIIII    DDDDDDDD    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
UU      UU  EEEEEEEEEE  TTTTTTTTTT    IIIIII    DDDDDDDD    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EEEEEEEE        TT          II      DD      DD  BBBBBBBB    DD      DD  EEEEEEEE    FFFFFFFF
UU      UU  EEEEEEEE        TT          II      DD      DD  BBBBBBBB    DD      DD  EEEEEEEE    FFFFFFFF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF          ....
UU      UU  EE              TT          II      DD      DD  BB      BB  DD      DD  EE          FF          ....
UUUUUUUUUU  EEEEEEEEEE      TT        IIIIII    DDDDDDDD    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FF          ....
UUUUUUUUUU  EEEEEEEEEE      TT        IIIIII    DDDDDDDD    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FF          ....

  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

```
{
{ Version:      'V04-000'
{
{*****************************************************************************
{*                                                                           *
{*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
{*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
{*  ALL RIGHTS RESERVED.                                                     *
{*                                                                           *
{*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
{*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
{*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
{*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
{*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
{*  TRANSFERRED.                                                             *
{*                                                                           *
{*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
{*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
{*  CORPORATION.                                                             *
{*                                                                           *
{*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
{*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
{*                                                                           *
{*                                                                           *
{*****************************************************************************
{

        module  $UETIDBDEF;

/*++
/*
/* FACILITY:  UETP, Regression Tests and Performance Measurement
/*
/* ABSTRACT:
/*      Provide the definitions needed for various data structures shared
/*      among different programs.  Defined are the PO representations of
/*      VMS's cluster and local I/O database referenced by the UETP$CLSIODB
/*      routine.
/*
/*--
/*
/* AUTHOR:  Richard N. Holstein, CREATION DATE:  27-Sep-1982
/*
/* MODIFIED BY:
/*
/*      V03-003 RNH0003         Richard N. Holstein,    22-Dec-1983
/*              Add provision for PB$W_STATE, PB$B_RSTATE, PB$B_CBL_STS,
/*              PB$B_PO_STS and PB$B_PT_STS fields.
/*
/*      V03-002 RNH0002         Richard N. Holstein,    15-Jul-1983
/*              Define fields for DEVCHAR and DEVCHAR2 in UCB records.
/*
/*      V03-001 RNH0001         Richard N. Holstein,    04-Jan-1983
/*              Modify to follow more closely the VMS style of SDL files.
/*              Change integers to be unsigned.  Add some items corresponding
/*              to those in $SBDEF and a flag to skip the system block for
```

```
/*              the system which is going through the cluster database.
/***
```

```
/*      Record types for I/O database
/*
/* The record types defined below are used not only for identifying record
/* types in the PO database created by UETP$CLSIODB, but to dispatch in case
/* statements.  Therefore, the order is important.
/*
constant (
    NULL_RTYPE                  /* Record type of null record
    ,SID_RTYPE                  /* Record type of system block record
    ,PATH_RTYPE                 /* Record type of path block record
    ,DDB_RTYPE                  /* Record type of DDB record
    ,UCB_RTYPE                  /* Record type of UCB record
    ,MPM_RTYPE                  /* Record type of shared memory record
    ,END_RTYPE                  /* Record type for end of all records
    ) equals 0 increment 1 prefix UID$;

/*      Record definitions
/*
/* These definitions are used for the individual records that describe the
/* peripherals available to a system and for the flags that say which records
/* are to be returned.  There are seven kinds of record:  a system record,
/* a path record, a DDB record, a UCB record, an MPM record, an end record
/* and a null record.  The first five correspond to similar items one finds
/* when traversing VMS's I/O database.  The end record gives a convenient way
/* to end if one reads the local data structure sequentially.  The null record
/* is available to allow for various housekeeping features.  A generic record
/* is also defined to emphasize the fields that are the same in all records.
/*

/*
/* Generic fields in all records
aggregate UETIDB_GNRC structure prefix UIDGNRC$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_xxxx_RTYPE
end UETIDB_GNRC;

/*
/* Null record
aggregate UETIDB_NULL structure prefix UIDNULL$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_NULL_RTYPE
    constant FFREE equals .;    /* First free byte
end UETIDB_NULL;

/*
/* Store system block info
aggregate UETIDB_SID structure prefix UIDSID$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_SID_RTYPE
    PBFL longword unsigned;     /* Pointer to first path block
    SYSTEMID byte unsigned dimension 6; /* System id - SB$S_SYSTEMID long
    SWTYPE character length 4;  /* ASCII software type
    SWVERS character length 4;  /* ASCII software version
```

```
    SWINCARN quadword unsigned; /* Software incarnation #
    HWTYPE character length 4;  /* ASCII hardware type, blank filled
    HWVERS byte unsigned dimension 12; /* ASCII hardware version
    NODENAME character length 16; /* ASCIC SCS nodename
    DDB longword unsigned;      /* Pointer to first DDB on list
    constant FFREE equals .;    /* First free byte
end UETIDB_SID;

/*
/* Store path info
aggregate UETIDB_PATH structure prefix UIDPATH$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_PATH_RTYPE
    STATE word unsigned;        /* Virtual circuit state
    LPORT_NAME character length 4; /* Local port name
    RSTATE byte unsigned;       /* Remote port state
    CBL_STS byte unsigned;      /* Overall cable status
    P0_STS byte unsigned;       /* Path A status
    P1_STS byte unsigned;       /* Path B status
    constant FFREE equals .;    /* First free byte
end UETIDB_PATH;

/*
/* Store DDB info
aggregate UETIDB_DDB structure prefix UIDDDB$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_DDB_RTYPE
    UCB longword unsigned;      /* Pointer to first UCB
    NAME character length 1;    /* Variable length .ASCIC - DDB name
    constant FFREE equals .;    /* First possible free byte
end UETIDB_DDB;

/*
/* Store UCB info
aggregate UETIDB_UCB structure prefix UIDUCB$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_UCB_RTYPE
    NUMBER word unsigned;       /* Unit number
    DEVCLASS byte unsigned;     /* Device class
    DEVTYPE byte unsigned;      /* Device type
    DEVCHAR longword unsigned;  /* First set of device characteristics
    DEVCHAR2 longword unsigned; /* Second set of device characteristics
    constant FFREE equals .;    /* First free byte
end UETIDB_UCB;

/*
/* Store shared (multiport) memory info
aggregate UETIDB_MPM structure prefix UIDMPM$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_MPM_RTYPE
    NUMBER word unsigned;       /* Memory unit number
    NAME character length 1;    /* Variable length .ASCIC - MPM name
```

```
    constant FFREE equals .;    /* First possible free byte
end UETIDB_MPM;

/*
/* End of records record
aggregate UETIDB_END structure prefix UIDEND$;
    FLINK address;              /* Pointer to next record of this type
    SIZE word unsigned;         /* Length of this record
    TYPE byte unsigned;         /* Always UID$K_END_RTYPE
    constant FFREE equals .; /* First possible free byte
end UETIDB_END;

/*      Flags
/*
/* Flags determining which subset of record types should be returned.  The
/* flags are not totally independent, i.e., there are some semantics needed
/* to determine which affect others.  If a data structure is "dependent"
/* (pointed to) by another kind of data structure, then returning information
/* about the first depends on returning information about the second.
/* Examples:  to return UCB info, one must return DDB info; to return path
/* block info one must return cluster info.  Note that the DDB flag is
/* redundant for local device info but necessary for cluster info.
/*
aggregate UETIDB_FLAGS structure prefix UIDFLAG$;
    SID bitfield mask;          /* If set, return system block info
    PATH bitfield mask;         /* If set, return path block info
    DDB bitfield mask;          /* If set, return DDB info
    UCB bitfield mask;          /* If set, return UCB info
    MPM bitfield mask;          /* If set, return shared memory info
    MYSYS bitfield mask;        /* If set, return cluster info about myself
end UETIDB_FLAGS;

end_module $UETIDBDEF;
```

UETIDBDEF
SDL

UETDMPF00
MAP

UETTTYS00
MAP

UETNRMS01
MAP

SATSSF18
MAP

UETFORT01
MAP

SATSSS10
MAP

UETFORT03
MAP

UETTAPE00
MAP

TCNTRLDEF
SDL

UETDISK00
MAP

UETUNT
SDL

UETNETS00
MAP

UETP

SATSSF17
MAP

UETDRIVW00
MAP

UETPHAS00
MAP

UETUNAS00
MAP

SYSTSTMAC
MAR

SATSSS09
MAP

UETFORT02
MAP

UETCOMS00
MAP

UETLBAK00
MAP